hangout: Sprint #5 Presentation

Antonio | Brandon | Eric | Ashley | Dean

Intro

No updates to Business Requirements Document

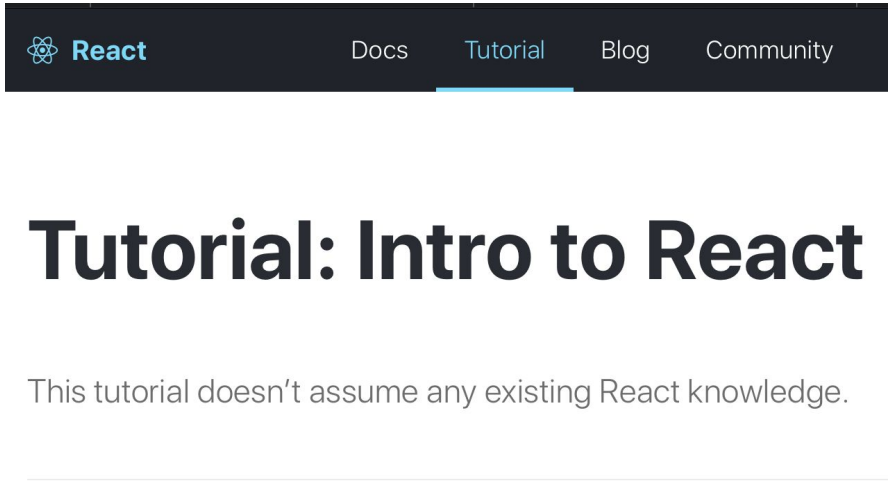
Management plan updates: Sprint board, project tracking matrix, burndown chart, sprint retrospective

No additional user stories released this sprint

We have a total of 2 features right now, user sign-up and user sign-in

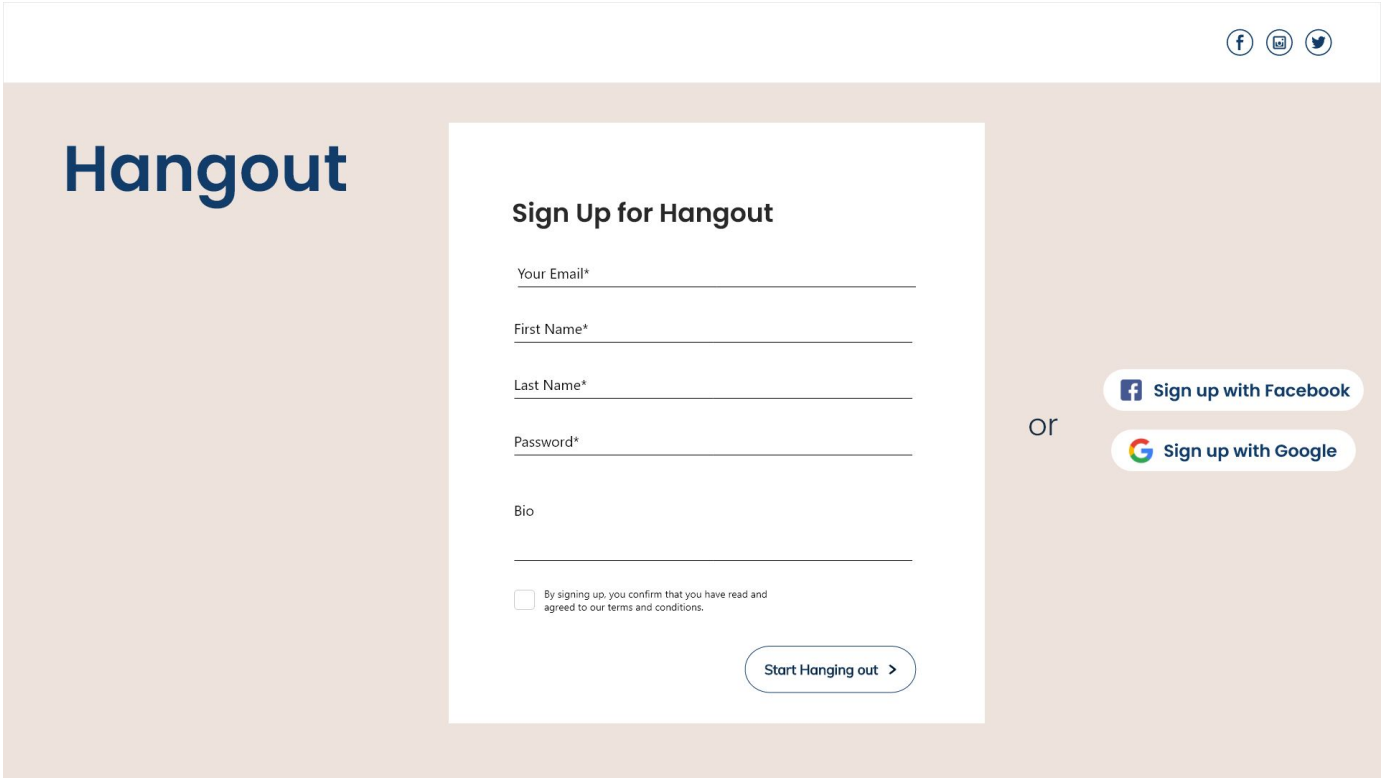
Where are we now?

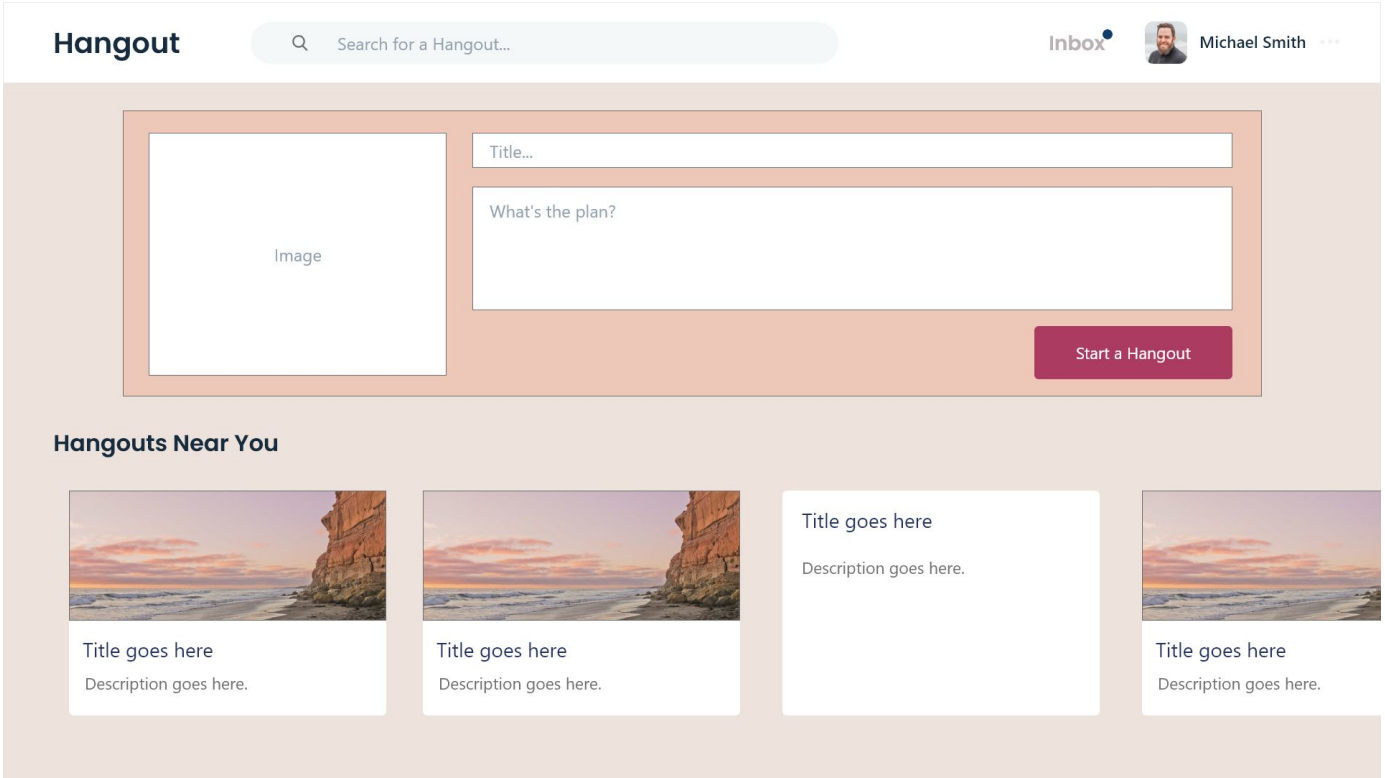
- Now that Hangout is hosted and is capable of creating and signing into user accounts, Hangout's development needs have quickly shifted towards front end development.
- This shift has caused some of to need to quickly learn ReactJS.



- Due to the sudden shift, lack of experience with ReactJS, and the time it takes to learn ReactJS, the majority of our time this Sprint was invested in learning ReactJS.

UI Design


Hangout
Sign Up for Hangout
Your Email*
First Name*
Last Name*
Password*
Bio
By signing up, you confirm that you have read and agreed to our terms and conditions.
Start Hanging out >
or
Sign up with Facebook
Sign up with Google


Hangout
Search for a Hangout...
Inbox
Michael Smith
Image
Title...
What's the plan?
Start a Hangout
Hangouts Near You
Title goes here
Description goes here.
Title goes here
Description goes here.
Title goes here
Description goes here.
Title goes here
Description goes here.

Goals of design

- Sleek, useable design. No nonsense.

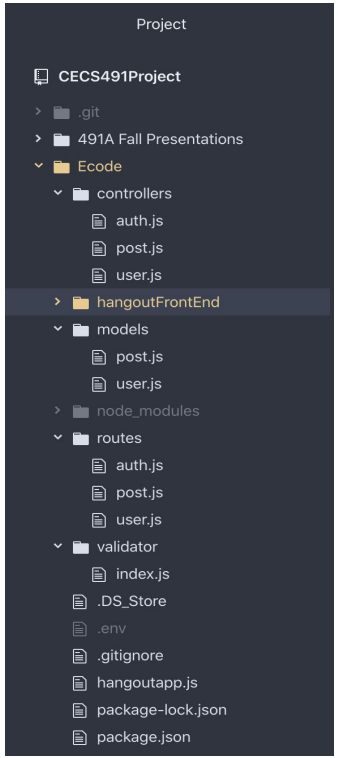
- Clear goal: easy access to find hangouts near you that you are interested in.

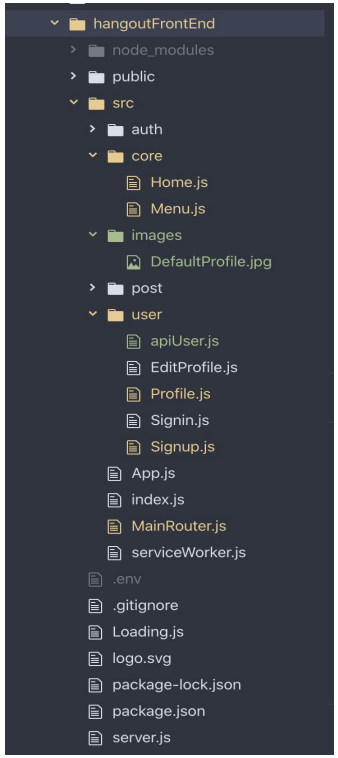
- Streamlined access: never more than two clicks away from seeing a Hangout™

- Reimagining social media: less clutter, more access to information you want to see.

-Co-opetition: By allowing for Facebook login, we have a convenient way to co-exist with one of our larger competitors.

Architecture





ODM used: Mongoose :

- Mongoose is an Object Data Modeling (ODM) library for MongoDB and Node. js.
  - stores data with JSON documents
  - speeds up application development and reduces the complexity of deployments
  - Mongoose objects are stored as schemas

Servers Used: 2

- Ubuntu server for hosting site on DigitalOcean.com
- MongoDB Server to store user information

Third Party API used: Axios

- Used the third party add on, axios-mock-adaptor, to generate mock requests to test code

Code Refactoring

- Menu bar refactored to combine home/signup page
  - Tabs were separate to make sure back end calls were made correctly.
  - Reducing the number of clicks a user has to navigate by adding redirects to a component.

- Json authentication moved to auth.js in Controllers folder

JSON Web Tokens

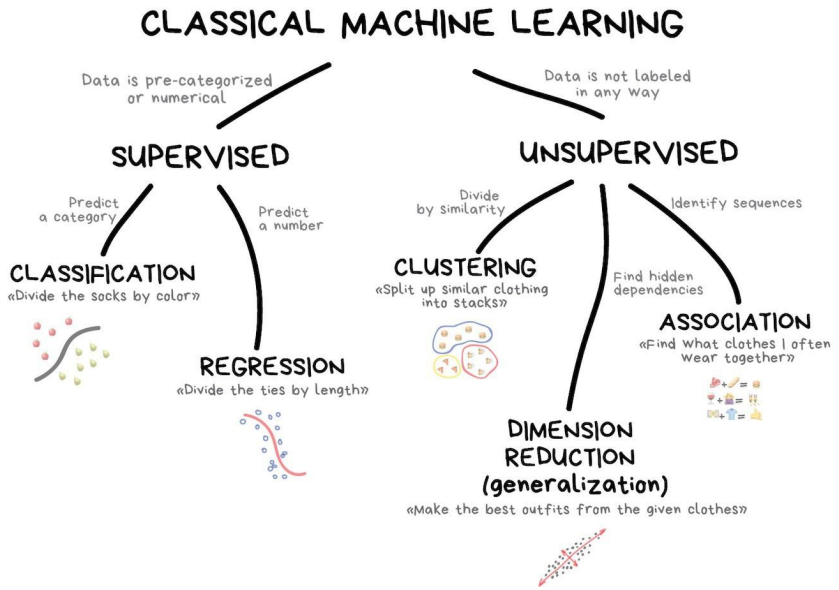
- Internet standard for creating access tokens to validate claims
- Proves that users are authenticated and that every interaction between other users and the webapp are legitimate

```
exports.signin = (req,res) => {
   //Find user based on email
   const {email, password} = req.body
   User.findOne({email}, (err, user) => {
      if(err || !user) {
         return res.status(401).json({
            error: "Email not registered. Please Sign up."
         })
      }
      //once user is found, password has to match
      if(!user.authenticate(password)) {
         return res.status(401).json({
            error:  "Email and password does not match!"
         });
      }
      const token = jwt.sign({_id: user._id}, process.env.JWT_SECRET);
      res.cookie("t",token, {expire: new Date() + 9999})
      const {_id, name, email} = user
      return res.json({token, user: {_id, email, name}});
   });
};
```

A decision tree.

Features: MMO, VS, Volleyball, Tennis
Swimming, Jogging

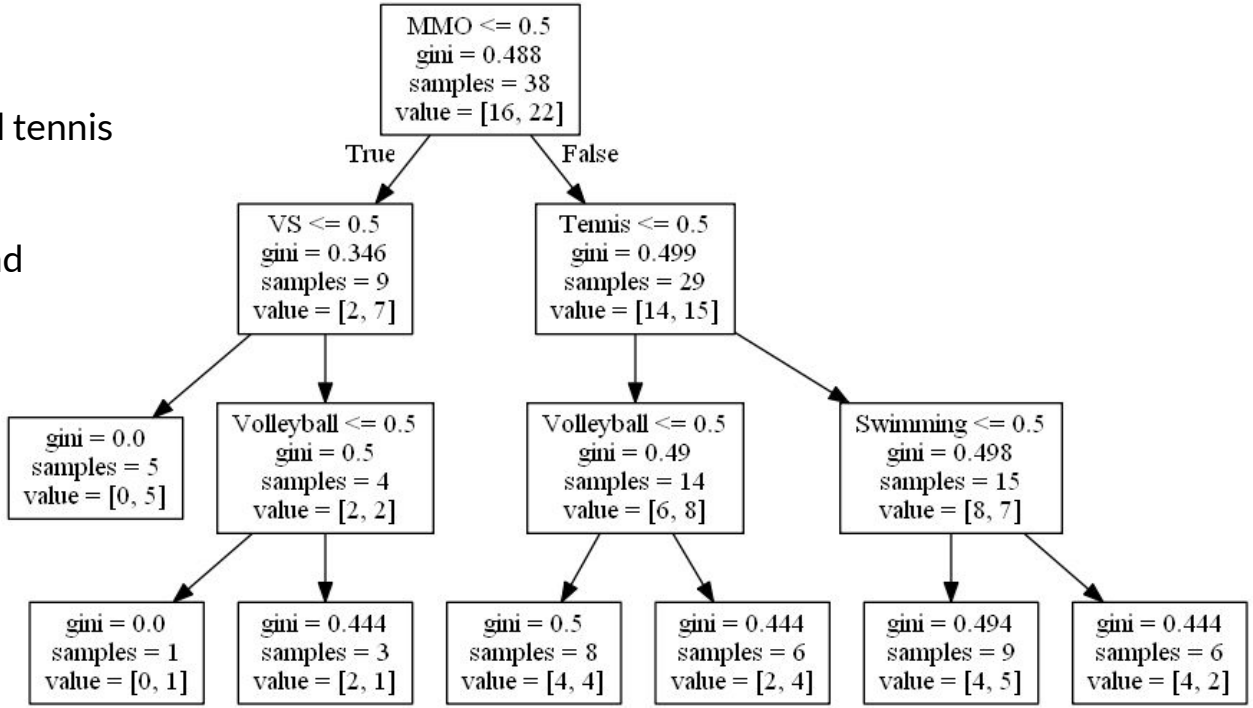
If time permits, might strengthen the model.



Illustrated Relationship:

An interest in MMOs and tennis → checks swimming.

More likely to recommend jogging if no interest in swimming.



The target label is jogging.

Training Data

The team plans to generate the data.

Technical Approach:

- Real instances and statistics.

Deployment

A web API using the Flask framework.

Alternatively, the team considered writing onto the server. We chose an API because the alternative can be a resort if something goes awry.

Sprint Goal

Our goal this sprint is to build out the back-end so that we can implement support for more features, and learn front end technologies.

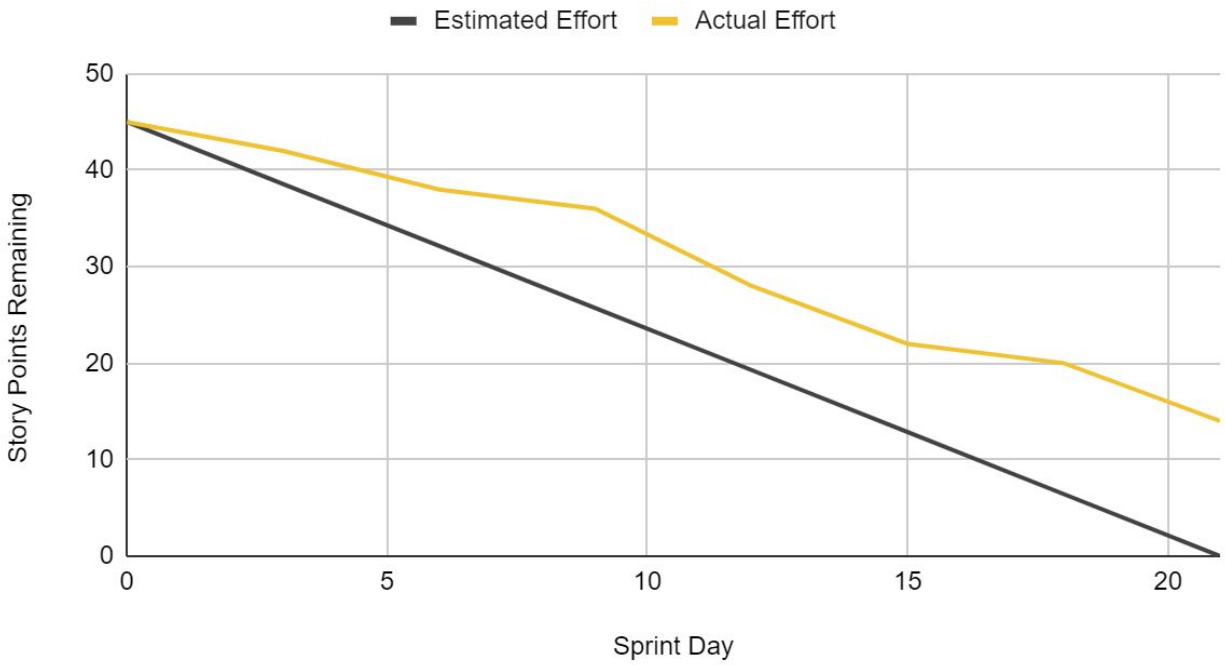
45 User Story Points Planned

31 User Story Points Achieved

User stories categorized: High, Medium, Low

Approximate # of “In-Progress”: 6

Sprint #5 Burndown Chart



Sprint Retrospective

Went well:
- Great effort from all team members
- Development research and learning

Could be better:
- Communication
- Improving Version Control Implementation

Next Sprint:
- Close out more user stories
- Focus on front-end development, matching front end features to what we've developed in back end